Deepfuse: Detecting Low Surface Brightness Galaxies with Convolutional Neural Networks

Marcos Tidball, Cristina Furlanetto, Rodrigo Flores de Freitas, Ana Chies Santos, Marco Antônio Canossa Gosteinski, William Schoenell
Instituto de Física, Universidade Federal do Rio Grande do Sul, Brazil

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

1 - INTRODUCTION

Low Surface Brightness Galaxies (LSBGs) are an important fraction of the galaxy population, having a fundamental role in the study of galaxy formation and evolution. According to Prole et al. (2018), a significant portion of the local Universe's matter content may be contained in diffuse sources, such as these galaxies. There is a special type of LSBGs that have recently sparked a lot of interest: Ultra-Diffuse Galaxies (UDGs) **(Fig. 1)**. One of the many questions that is still under debate about these objects is how they form. Luckily, various recent astronomical surveys have cameras capable of detecting low surface brightness features. However, huge amounts of data are produced, making visual inspection virtually impossible, while also being subject to human failure.

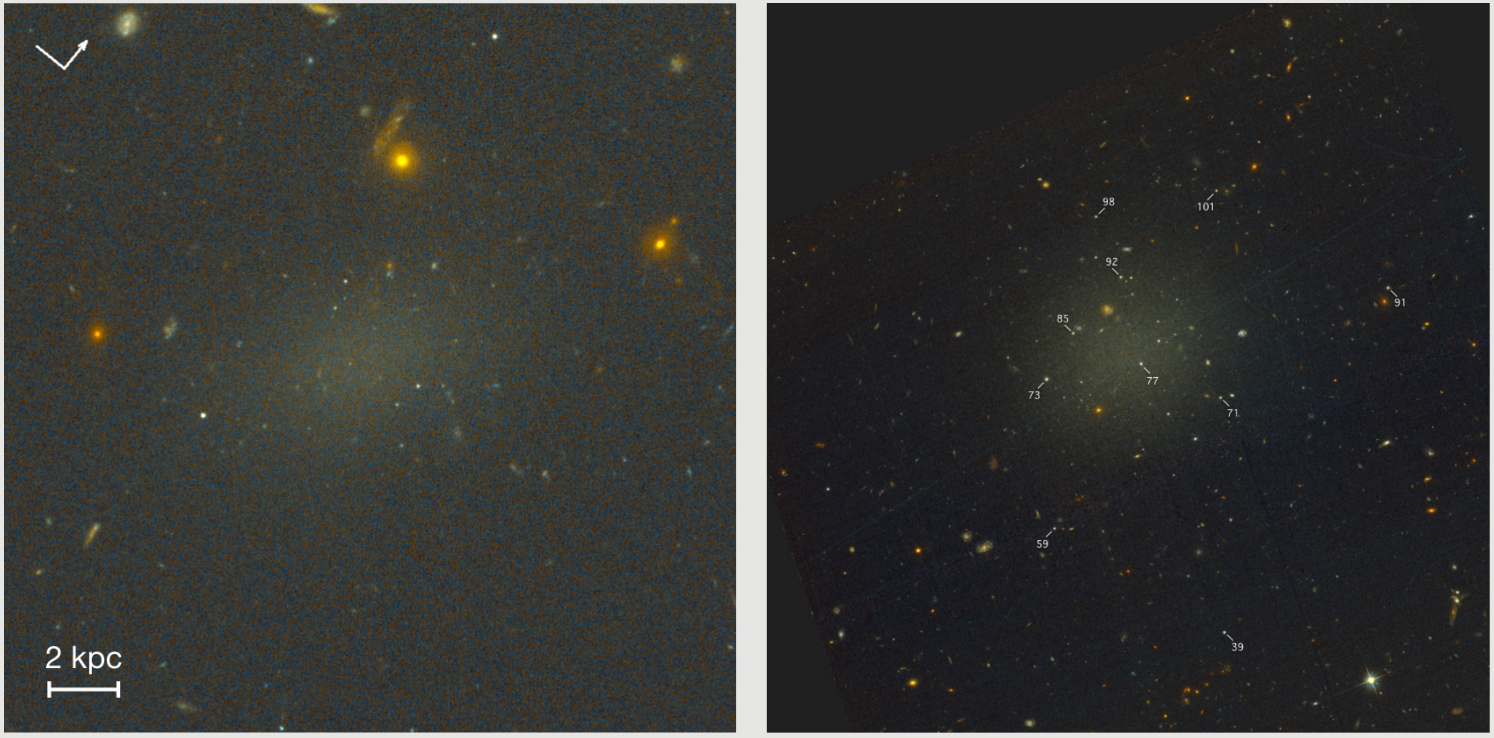

Fig. 1: Left: Dragonfly 44 (van Dokkum et al. 2017); Right: NGC1052–DF2 (van Dokkum et al. 2018)

Deep learning based frameworks have recently surpassed human accuracy in tasks related to visual classification of objects. This was made possible thanks to the advent of Convolutional Neural Networks (CNNs) and neural network architectures that are able to have a large number of layers. This project aims to **automatically detect LSBG candidates in large area astronomical images and train a neural network to classify these candidates**, indicating whether they are low surface brightness galaxies or not.

2 - CONVOLUTIONAL NEURAL NETWORKS

The basis of a CNN is the convolution operation. This operation revolves around passing a filter through an image. The filter is a grid of values that marches along the image and multiplies the values of each pixel around the grid area by the values in the grid **(Fig. 2)**. These values are then added together to form the value of the pixel on the new post-convolution image. Depending on the values of the filter, also called kernel, it is possible to identify different characteristics of the image such as curvature, horizontal lines, etc.

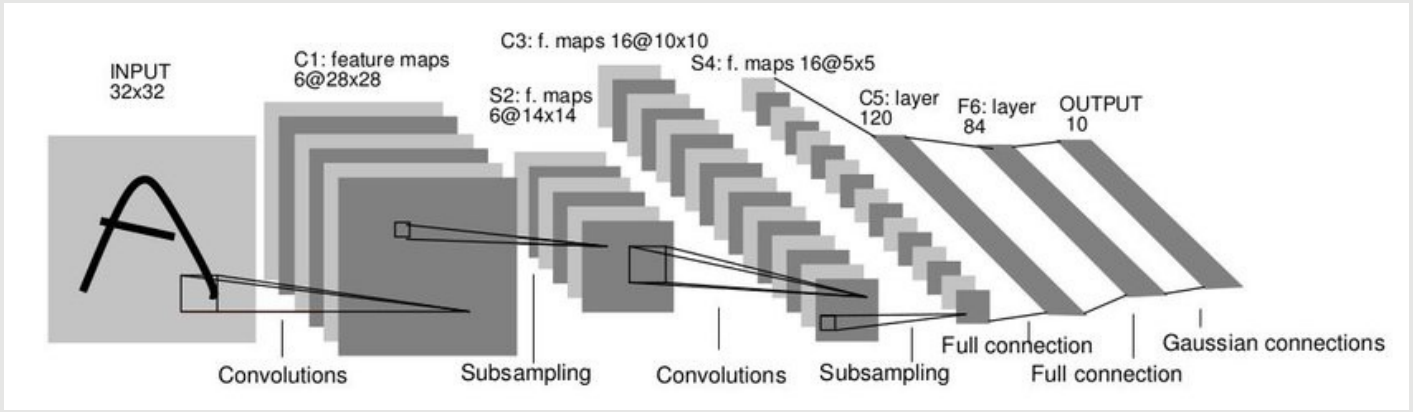

Fig. 3: A convolutional neural network. We can notice that as we go deeper the image gets smaller and starts to have more channels. Each channel represents a different filter that is applied. After a certain point the smaller image with all its many channels is flattened into a 1D vector, which then has the values for the predicted labels. (LeCun et al. 1999)

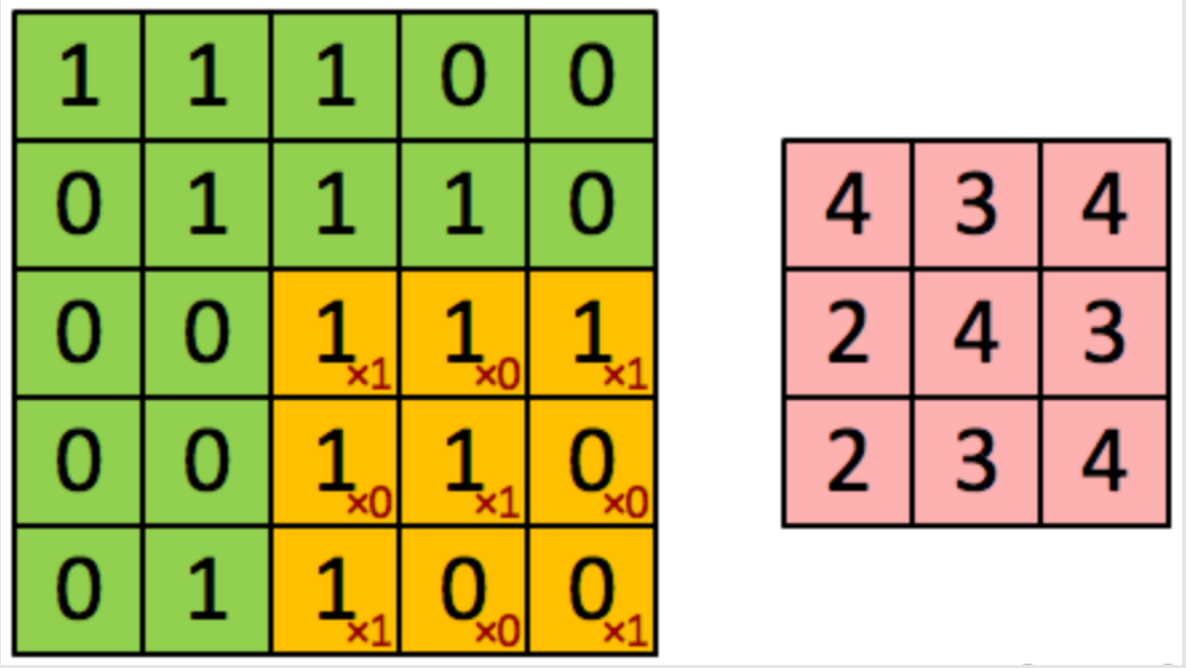

Fig. 2: Convolution operation: large green square on the left is an image, small yellow square on the left is a kernel and red square on the right is the convolved image. (http://deeplearning.stanford.edu/tutorial/)

In a CNN, multiple filters will be convolved with the image, each of them extracting different features. The deeper the CNN the more filters are used and therefore more different features are extracted. What makes a CNN special is its ability to learn the best filter values according to its training data. This is possible because on each layer many convolutions happen with different filters, with a nonlinear function being applied afterwards: this allows for more complex data representations to be learned **(Fig. 3)**.

In order for the network to learn these filters it needs to be trained. In this process, the network is fed with labelled images. Initially the CNN does not have access to the real labels of the images and has to guess them. Afterwards, a loss function is used to calculate how wrong was this prediction. Through a process called "backpropagation" it is possible to use the value of the loss function to change the filter values of the network's different layers. Therefore, **it is possible to learn the best filter values for the classification task.**

3 - DEEPFUSE

The pipeline created to automatically detect LSBGs is divided into two parts: the first uses a **source extractor** tool called DeepScan (Prole et al. 2018), specialized to detect low surface brightness features on astronomical images. The second part then implements a **CNN to classify these extracted sources**, classifying whether they are LSBGs or not. This pipeline was then called Deepfuse. It is described more thoroughly in the following sections.

3.1 - DEEPSCAN

The DeepScan library detects low surface brightness objects in astronomical images without fragmenting the source structure. It works thanks to an algorithm that gradually identifies neighbouring pixels, grouping them based on the density of pixels above a signal-to-noise threshold within a certain distance. This process goes on until there are no more similar pixels close enough for them to be considered part of the same source **(Fig. 4)**. From this, it is possible to directly detect astronomical sources in images.

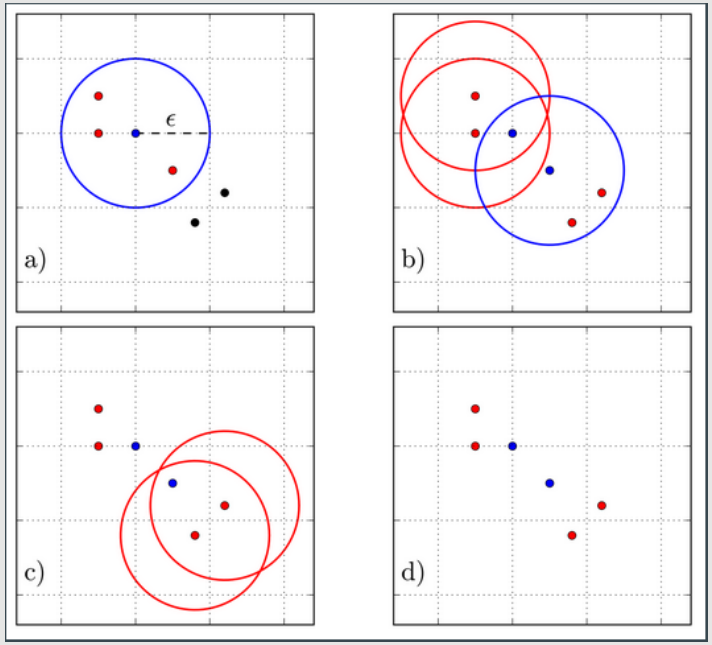

Fig. 4: The DeepScan nearest neighbours algorithm in action. It first detects a core point (blue), and then proceeds to check if the secondary points (red) are also core points. When there are no more points to be added the cluster is completed. (Prole et al. 2018)

While the algorithm is specialized to detect low surface brightness features, applying it to an image gives a large number of detections, ranging from very faint to very bright objects. Since our search is for a very diffuse kind of object though, with very low surface brightness, we implemented various cuts in different physical quantities. The idea behind this is to fine-tune DeepScan for the detection of LSBGs, disregarding other objects in order to make the classification process by the CNN faster, with less images to be analysed. For a DECam pointing, this process has reduced the number of detected objects by three orders of magnitude **(Fig. 5)**. Each detected source is cropped out of the image and saved as a stamp, which is then going to be used by our CNN.

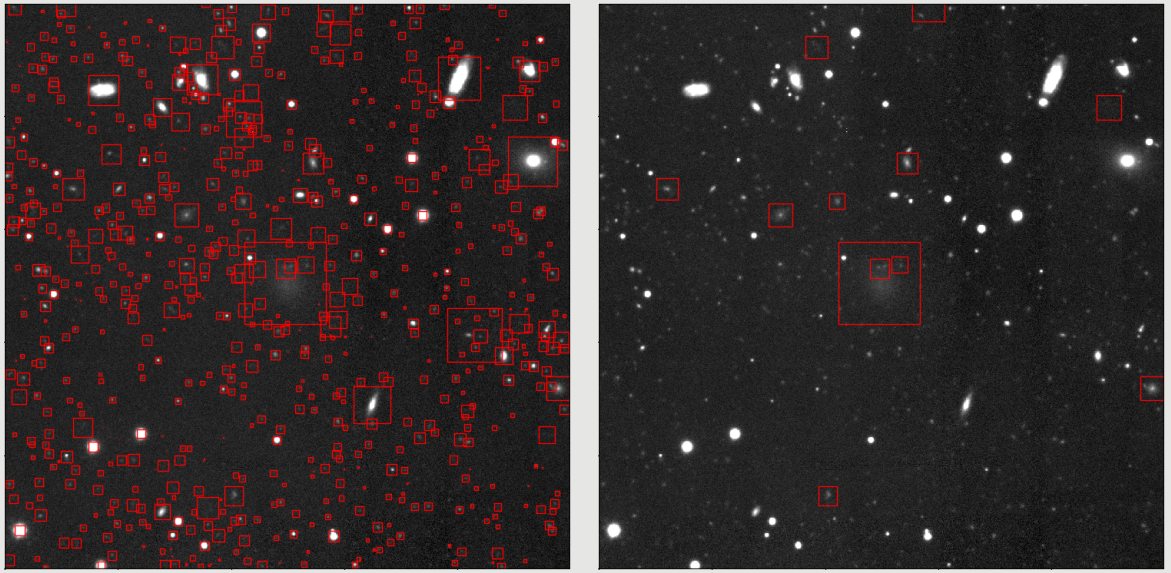

Fig. 5: DeepScan detections before (left) and after (right) applying the threshold cuts in surface brightness and effective radius. This image is was obtained by DECam at Blanco Telescope, with one LSBG candidate located in the centre.

3.2 - THE NEURAL NETWORK

In order to train the CNN we needed two classes of images: positives (LSBGs) and negatives (non-LSBGs). For the positive sample we used simulated LSBGs images due to the small amount of real data available. These simulations were made assuming magnitude, surface brightness and effective radius with distributions according to data available in the literature. For the negatives, on the other hand, we extracted stamps from DeepScan detections in astronomical images. These were visually inspected and considered as not containing LSBGs. The simulated galaxies were inserted in the same astronomical images from where we extracted the non-LSBGs. This way it is possible to have the same image properties for both samples. **(Fig. 6)**.

Since we currently have only a small amount of both positive and negative images, we made use of transfer learning: using a CNN that was pre-trained on large amounts of data and retraining just part of it on new data. This makes it possible to maintain the filters that detect general features, without the network being impaired due to the small amount of training samples, while also specializing more easily on new data.

4 - NEXT STEPS

The project is still ongoing. We are currently performing tests with different networks and parameters, as well as testing different cutout sizes for the sources extracted from DeepScan in order to find the combination with the best results.

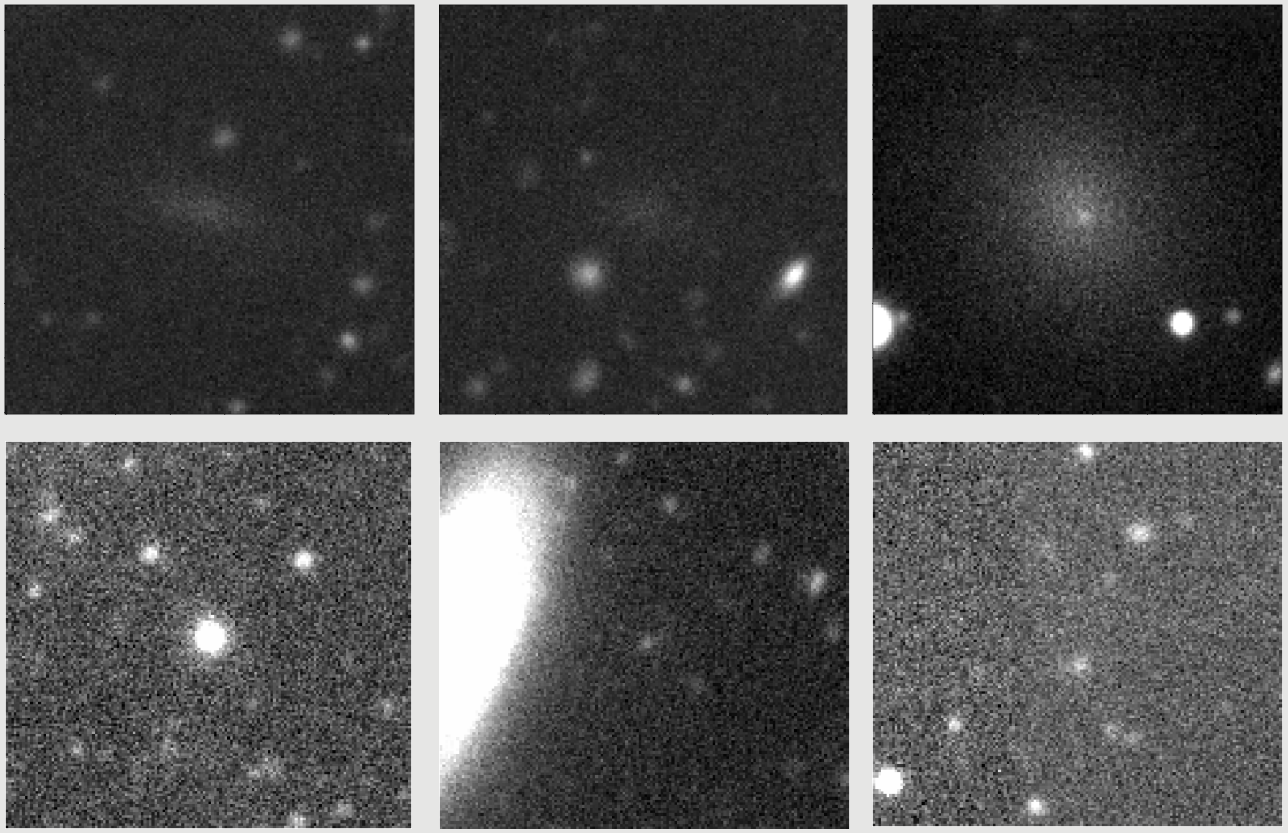

Fig. 6: Images used for training. The top row is composed of positive images, i.e. simulated LSBGs. The bottom row is composed of negatives, i.e. sources extracted from DeepScan that were considered as being non-LSBGs.

REFERENCES

Prole, D. J., Davies, J. I., Keenan, O. C., & Davies, L. J. M. 2018, MNRAS, 478, 667.

van Dokkum, P., Danieli, S., Cohen, Y., et al. 2018, Nature, 555, 629.

van Dokkum, P., Abraham, R., Romanowsky, A. J., et al.2017, ApJL, 844, L11.

ACKNOWLEDGEMENTS

We thank Clécio de Bom, Patrick Schubert and Luciana Dias for technical support related to the CNN.

We thank Áttila Leães Rodrigues for computational time.

We also thank CNPq and FAPERGS for funding this research.